AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF CLAIMS:

Claims 1-64. (Cancelled)

65. (New) A compiler, disposed on a computer readable medium, the compiler including instructions to cause a first processor to:

access source code expressed in a computer language having an instruction set that includes:

an instruction set statement to declare a network protocol, the statement having a syntax to name packet data at one or more specified locations within a network packet; and

an instruction set statement to specify a rule having a syntax comprising a Boolean expression and at least one action to perform if the Boolean expression is true; and

based on the source code, output instructions to cause a second processor to process received network packets by:

assigning value(s) to the named packet data based on data within a network packet in accordance with the instruction set statement to declare a network protocol; and

applying at least one rule in accordance with the instruction set statement to specify a rule.

66. (New) The compiler of claim 65, wherein the source code comprises multiple instruction set statements to specify multiple rules and where the output instructions

cause the second processor to apply the rules in an order corresponding to the order in which the statements to specify the multiple rules appear in the source code.

67. (New) The compiler of claim 66, wherein the at least one action to perform returns a value controlling whether a subsequent one of the multiple rules is applied.

68. (New) The compiler of claim 65,

wherein the instruction set statement to declare a network protocol comprises a statement having a syntax comprising of:

protocol protocol_name { [field definitions(s)] }

and wherein the syntax to name packet data comprises field definitions having a syntax comprising of:

field_name {protocol_name [offset : field size] }.

69. (New) The compiler of claim 65,

wherein the syntax of the instruction set statement to declare a network protocol further comprises at least one declaration of an encapsulated packet header associated with another network protocol; and

wherein the output instructions comprise instructions to name data of the encapsulated packet header.

70. (New) The compiler of claim 69, wherein the at least one declaration of the encapsulated packet header comprises a declaration including an evaluation expression identifying whether the encapsulated packet header is present in a network packet.

71. (New) The compiler of claim 70, wherein the declaration of the encapsulated packet header comprises a declaration having a syntax comprising of:

demux { Boolean_expression {protocol_name at offset} }

wherein the protocol_name identifies a protocol declared by an instruction statement to declare a network protocol.

72. (New) The compiler of claim 65, wherein the instruction set includes:

an instruction set statement to name a set of values, the set of values comprising a set of tuples, wherein each tuple comprises one or more key values; and

an instruction set search statement to search the set of values.

73. (New) The compiler of claim 72, wherein the instruction set search statement comprises a syntax of:

set_name.search_name(key(s)),

wherein the search_name is an expression that evaluates to true when at least one tuple of the set of values identified by the set_name matches the value(s) of the key(s) specified.

74. (New) The compiler of claim 72, wherein the set of values are set by instructions external to the instructions output in response to the source code.

75. (New) The compiler of claim 72, wherein the values are set based on the received network packets.

76. (New) A method, comprising:

accessing source code expressed in a computer language having an instruction set that includes:

an instruction set statement to declare a network protocol, the statement having a syntax to name packet data at one or more specified locations within a network packet; and

an instruction set statement to specify a rule having a syntax comprising a Boolean expression and at least one action to perform if the Boolean expression is true; and

based on the source code, outputting instructions to cause a processor to process received network packets by:

assigning value(s) to the named packet data based on data within a network packet in accordance with the instruction set statement to declare a network protocol; and

applying at least one rule in accordance with the instruction set statement to specify a rule.

77. (New) The method of claim 76, wherein the source code comprises multiple instruction set statements to specify multiple rules and where the output instructions cause the second processor to apply the rules in an order corresponding to the order in which the statements to specify the multiple rules appear in the source code.

78. (New) The method of claim 77, wherein the at least one action to perform returns a value controlling whether subsequent one of the multiple rules is applied.

79. (New) The method of claim 76,

wherein the instruction set statement to declare a network protocol comprises a statement having a syntax comprising of:

protocol protocol_name { [field definitions(s)] }

and wherein the syntax to name packet data comprises field definitions having a syntax comprising of:

field_name {protocol_name [offset : field size] }.

80. (New) The method of claim 76,

wherein the syntax of the instruction set statement to declare a network protocol further comprises at least one declaration of an encapsulated packet header associated with another network protocol; and

wherein the output instructions comprise instructions to name data of the encapsulated packet header.

81. (New) The method of claim 80, wherein the at least one declaration of the encapsulated packet header comprises a declaration including an evaluation expression identifying whether the encapsulated packet header is present in a network packet.

82. (New) The method of claim 81, wherein the declaration of the encapsulated packet header comprises a declaration having a syntax comprising of :

demux { Boolean_expression {protocol_name at offset} }

wherein the protocol_name identifies a protocol declared by an instruction statement to declare a network protocol.

83. (New) The method of claim 76, wherein the instruction set includes:

an instruction set statement to name a set of values, the set of values comprising a set of tuples, wherein each tuple comprises one or more key values; and

an instruction set search statement to search the set of values.

84. (New) The method of claim 83, wherein the instruction set search statement comprises a syntax of:

set_name.search_name(key(s)),

wherein the search_name is an expression that evaluates to true when at least one tuple of the set of values identified by the set_name matches the value(s) of the key(s) specified.

85. (New) The method of claim 84, wherein the set of values are set by instructions external to the instructions output in response to the source code.

86. (New) The method of claim 85, wherein the values are set based on the received network packets.